(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 105550977 A
(43)申请公布日 2016.05.04

(21)申请号 201610066304.7

(22)申请日 2016.01.29

(71)申请人 中国人民解放军国防科学技术大学
地址 410073 湖南省长沙市开福区德雅路109号

(72)发明人 刘世永 李军 吴秋云 熊伟 陈荦 钟志农 吴烨

(74)专利代理机构 北京中济纬天专利代理有限公司 11429
代理人 陈立新

(51)Int.Cl.
G06T 1/20(2006.01)
G06T 7/00(2006.01)
G06F 17/30(2006.01)

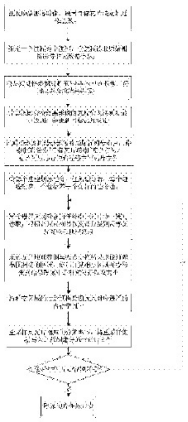
权利要求书2页 说明书8页 附图4页

(54)发明名称

一种并行方式栅格影像切片方法

(57)摘要

本发明属于地理空间信息处理技术领域，涉及一种并行方式栅格影像切片方法。包括步骤：第一步：获取原始栅格影像，设置目标瓦片级别和进程总数；第二步：读取原始栅格影像的元数据信息；第三步：将原始栅格影像变换至WebMercator投影；第四步：计算投影变换结果影像的瓦片最大级别和最小级别；第五步：计算投影变换结果影像的瓦片行列号范围，建立空瓦片文件；第六步：采用车轮法，给每个进程划分任务；第七步：每个进程分别对各自任务池中瓦片逐一进行读取，并反算瓦片对应的地理范围；求相交的区域在投影变换结果影像中的相对位置以及大小，将相交区域的原始栅格数据读内存空间中；第八步：重采样后写入之前创建好的空瓦片文件中。



CN 105550977 A

1.一种并行方式栅格影像切片方法，其特征在于，包括以下步骤：

第一步：获取原始栅格影像，设置目标瓦片级别和进程总数；

第二步：指定一个进程为主进程，主进程读取原始栅格影像的元数据信息；

第三步：将原始栅格影像变换至WebMercator投影，得到投影变换结果影像；

第四步：计算投影变换结果影像的瓦片最大级别和最小级别；若设置的目标瓦片级别大于最大级别，则将最大级别作为目标瓦片级别，若设置的目标瓦片级别小于最小级别，则将最小级别最为目标瓦片级别；建立文件夹用于输出存放瓦片数据，并命名为目标瓦片级别；

第五步：计算投影变换结果影像的瓦片行列号范围；并按照路径为/目标瓦片级别/瓦片列号/瓦片行号.png的路径建立空瓦片文件；

第六步：根据步骤五中的瓦片行列号范围，并以瓦片为单位，采用车轮法，给每个进程划分任务，任务划分后，每个进程对应一个包括若干个瓦片的任务池；

第七步：每个进程分别对各自任务池中瓦片逐一进行读取，根据瓦片行列号以及瓦片级别反算瓦片对应的地理范围；求解瓦片地理范围与投影变换结果影像地理范围相交的区域，然后计算相交区域在投影变换结果影像中的相对位置以及大小，利用GDAL类库中的RasterIO函数将相交区域的原始栅格数据读入对应进程的内存空间中；

第八步：将步骤七中各进程读入相交区域的原始栅格数据重采样到瓦片相应的分辨率下；将重采样数据写入第五步中创建好的空瓦片文件中。

2.如权利要求1所述的一种并行方式栅格影像切片方法，其特征在于，所述第四步中计算投影变换结果影像的瓦片最大级别和最小级别具体过程为：假设以像素为单位每个瓦片的大小为tilesize×tilesize，投影变换结果影像长为Xsize、宽为Ysize，以米(m)为单位投影变换结果影像分辨率为reslt，Resolution(h)表示其第h级别的瓦片分辨率，Resolution(h)通过以下公式计算得出Resolution(h)=(2×π×R)/(tilesize×2^h)，R为地球半径等于6378137m，h为整数；投影变换结果影像的瓦片最大级别tmaxz为Resolution(h)最接近reslt但不大于reslt时的h值；假设$minRes=\left(reslt\times max(Xsize,Ysize)\right)/tilesize$，max为取最大值函数，则瓦片最小级别tminz为Resolution(h)最接近minRes但不大于minRes时的h值。

3.如权利要求1所述的一种并行方式栅格影像切片方法，其特征在于，所述第五步计算行列号的具体过程为：假设投影变换结果影像地理范围为[ominX,ominY,omaxX,omaxY]，其中该范围为一个矩形四边形，ominX,omaxY为左上角点坐标，omaxX,ominY为右下角点坐标；瓦片行列号范围为[tminX,tminY,tmaxX,tmaxY]，其中该范围为一个矩形四边形，tminX,tmaxY为左上角行列号，tmaxX,tminY为右下角行列号，则其之间满足以下映射关系：tminX=ceil((ominX+originShift)/(Resolution×tilesize))-1；tminY=ceil((ominY+originShift)/(Resolution×tilesize))-1；tmaxX=ceil((omaxX+originShift)/(Resolution×tilesize))-1；tmaxY=ceil((omaxY+originShift)/(Resolution×tilesize))-1；其中originShift=(2×π×6378137)/2，即地球周长的一半，Resolution=(2×π×6378137)/(tilesize×2^{level})，tilesize为瓦片边长大小，ceil表示向上取整。

4.如权利要求1所述的一种并行方式栅格影像切片方法，其特征在于，所述第六步任务

划分的具体方法为:假设rank(i)表示第i个进程,则属于rank(i)任务池中的瓦片其瓦片行列号[tx,ty]满足tx=tminX+(pos%twidth);ty=tmaxY-(pos/twidth),其中twidth=tmaxX-tminX,theight=tmaxY-tminY,pos=j×n+i,(j∈[0,1…,k]),j为整数,$k=\left\lfloor\frac{tcount}{n}\right\rfloor$,⌊ ⌋表示向下取整,tcount=twidth×theight,n表示进程总数,i为当前进程的进程号,tminX,tminY,tmaxX,tmaxY为第五步中所求的瓦片最小最大行列号;如果当tcount%n≠0时,进程号i<tcount%n的部分进程还需处理瓦片行列号[tx_2,ty_2]满足以下条件的瓦片:tx_2=tminX+(pos_2%twidth);$ty_2 = tmaxY - \lfloor(pos_2/twidth)\rfloor$,其中$pos_2$=tcount-tcount%n+i。

5.如权利要求1所述的一种并行方式栅格影像切片方法,其特征在于,所述第七步的具体过程为:假设rank(i)表示第i个进程,rank(i)当前处理的瓦片行列号为[tx_i,ty_i],首先进程rank(i)在先前瓦片输出目录中的level文件夹内检查是否已经存在名为tx_i的文件夹,如果不存在则在level目录下创建名为tx_i的文件夹;然后rank(i)根据瓦片行列号以及瓦片级别level反算瓦片对应的地理范围[gminX,gminY,gmaxX,gmaxY],单位为米,具体解算过程如下:gminX=tx_i×Resolution-originShift,gminY=ty_i×Resolution-originShift,gmaxX=(tx_i+1)×Resolution-originShift,gmaxY=(ty_i+1)×Resolution-originShift,其中Resolution=(2×π×6378137)/(tilesize×2^{level}),originShift=(2×π×6378137)/2;然后求解瓦片地理范围与投影变换结果影像地理范围相交的区域,假设相交区域为[gistminX,gistminY,gistmaxX,gistmaxY],单位为米,然后计算相交区域在投影变换结果影像中的相对位置以及大小,具体解算过程如下:以米为单位假设投影变换结果影像的地理范围为[gimgminX,gimgminY,gimgmaxX,gimgmaxY],空间分辨率为Geores,则$rxsize = \frac{gistmaxX - gistminX}{Geores}$,$rysize = \frac{gistmaxY - gistminY}{Geores}$当gistminX=gimgminX时rx=0,否则$rx = \frac{gistminX - gimgminX}{Geores}$;当gistmaxY=gimgmaxY时ry=0,否则$ry = \frac{gimgmaxY - gistmaxY}{Geores}$;其中rx,ry为投影变换结果影像中以左上角为原点的像素坐标系中的读取位置,rxsize和rysize为读取大小;而后利用GDAL类库中的RasterIO函数将rx,ry,rxsize,rysize填入相应参数位置,将相交区域的原始栅格数据读入rank(i)的内存空间中。

6.如权利要求1所述的一种并行方式栅格影像切片方法,其特征在于,所述第八步中重采样方法为最邻近内插法。

一种并行方式栅格影像切片方法

技术领域

[0001] 本发明属于地理空间信息处理技术领域，涉及一种并行方式栅格影像切片方法。

技术背景

[0002] 随着卫星传感器技术以及无人机航拍技术的快速发展，遥感影像的空间和时间分辨率都有大幅度地提高，单幅遥感影像文件的数据量也急剧增加。当前主流GIS软件以及互联网地图应用在WebGIS(网络地理信息系统)解决方案中都广泛采用地图切片(Tile，又称瓦片)的发布策略，这种方法通过预先将原始影像按照一定的切分规则切割成一张张大小相同的瓦片，以加载小数据量瓦片的方式达到在网络带宽受限的条件下实现影像的高效显示。但是目前的商业GIS软件其地图切片的生成以及发布成本较高，操作较复杂，以行业内知名的ArcGIS系列软件为例，要将栅格影像进行切片不仅要安装ArcGIS Desktop，而且还要安装庞大的ArcGIS Server，并且操作十分复杂繁琐，极大增加时间和物力成本。最重要一点是随着单幅遥感影像文件的分辨率以及数据量的急剧增加，其相应的切片数量会呈现几何级数式的增长，传统的串行算法以及商业GIS软件是通过单机预先切好瓦片，再对外提供，这种传统切片技术计算资源利用低下，并且没有错误恢复机制，一旦切片过程中出现故障，整个切片工作得推倒重来，无法在原有的进度上继续进行。因此在硬件性能高速发展的情况下，如何利用高性能计算技术，方便快捷并且高效快速地进行栅格影像切片，已经成为WebGIS地图应用中快速可视化方面必须要解决的重要问题。

[0003] 栅格影像切片是指将指定地理范围的栅格影像，在某一级别比例尺下，基于一定的切割准则切割成若干行和列的固定尺寸的正方形栅格图片的过程，这些规整的影像切片又称为瓦片。每个瓦片在该级别比例尺下都有一个独立的编码，该编码由瓦片的行列号以及比例尺的级别构成。当浏览器前端在影像浏览时，根据当前的地理范围动态地计算所需显示的影像切片的行列号，通过行列号以及当前比例尺级别得到瓦片的编码，而后通过编码向服务器端请求相应瓦片进行显示，从而达到快速响应目的。

[0004] 目前栅格影像并行切片方法主要有三种思路：一种是CPU(Central Processing Unit，中央处理器)+GPU(Graphic Processing Unit，图形处理单元)的方式，利用GPU的计算能力进行并行加速，这种方法并行能力受限于GPU硬件的能力，并行程度有限，而且会提高系统架构成本；另一种是利用分布式集群系统，将切片任务划分为多个子任务，各子任务在多个分布式节点上同时进行，这种方法可以比较方便将原有的串行方法并行化，但是当数据规模比较大的时候，前期的数据分布式存储以及后期的结果合并都比较耗时；再一种是基于多线程并行技术，各线程将任务划分成多个子任务，每个子任务同时进行，这种方法可以充分利用本机计算资源，但是由于线程间并行属于细粒度并行，各线程共享父进程的内存空间，容易造成系统的不稳定，并且这种方法可扩展性不是很好；目前基于MPI(Message Passing Interface，消息传递接口)的多进程方式进行栅格影像的并行切片研究较少，这种方法利用共享外存的高性能集群，可扩展性强，稳定性强，可以充分利用多机计算资源，由于集群之间共享外存，所以数据无需提前分布式存储，可以极大地提高影像切

片的效率。

发明内容

[0005] 本发明的目的在于提供了一种基于新思路的栅格数据并行切片方法。本发明利用多进程技术对任务进行划分，每个进程独立进行其相应切片工作，当某个进程出现问题时，其余进程仍可正常完成切片工作，互不影响。本发明切出的结果瓦片兼容目前绝大多数互联网地图应用，可以直接投入生产，并且本发明支持断点，一旦发生断电等现象，算法可在原有工作的基础上继续进行切片，避免了重复工作的问题。

[0006] 针对上述技术问题，本发明提供了一种并行方式栅格影像切片方法。步骤如下：

[0007] 第一步：获取原始栅格影像，设置目标瓦片的级别和进程总数。

[0008] 设定目标瓦片的级别level以及MPI参与的进程总数n，以及瓦片输出路径。MPI会为每一个进程分配一个进程号i($0 \leq i<n$，i取整数)，后续可通过进程号来达到对每个进程单独控制的目的。

[0009] 第二步：读取原始栅格影像的元数据信息。

[0010] 设置其中一个进程为主进程，主进程读取原始栅格影像的元数据信息，包括影像的长宽、波段数，投影坐标系、数据的无效值；

[0011] 第三步：将原始栅格影像变换至WebMercator投影。

[0012] 当前主流WebGIS软件以及互联网地图应用的切分瓦片坐标系统普遍采用Google提出的WebMercator投影坐标系，因此在进行切片前首先需将原始栅格影像的坐标系统转换至WebMercator目标投影系，然后基于WebMercator投影下的坐标进行切分。为了加快投影变换的效率，本发明采用以下策略：主进程根据之前读取的原始影像投影坐标系，如果原始栅格影像不是WebMercator投影系(在地理栅格影像中，所有投影坐标系都对应一个由开放地理空间联盟制定的WKT编码，利用GDAL库的相关函数从原始影像中读取该影像的WTK形式投影信息，通过比较原始影像的WKT编码和WebMercator投影的WTK就可以判断该影像的投影系是否为WebMercator投影)。主进程利用GDAL类库的GDALAutoCreateWarpedVRT函数，重采样方法采用最邻近内插法，将原始栅格影像变换至WebMercator投影坐标系下，得到投影变换结果影像，以vrt格式文件进行保存，后续的所有切片操作都是基于投影变换结果影像进行，并将原始栅格影像的无效值、影像宽度、波段数等元数据信息写入vrt文件中。vrt是一个xml格式的文件，本身不存储像素的色彩信息，它通过与原始栅格影像相关联，通过描述信息记录自身像素的坐标(自身像素坐标就是指像素点在投影变换结果影像像素坐标系下的坐标)与相关联栅格影像的映射关系，以及投影变换结果影像的长宽、波段数等元数据信息，当通过vrt文件读取栅格数据时，首先基于映射关系，找到其在原始栅格影像文件中的位置，然后从原始栅格影像文件相应位置中把栅格数据读出。由于vrt文件只记录一些描述性信息，包括投影变换结果影像的长宽、波段数、像素点的数据类型、投影坐标系，以及与原始栅格影像的映射关系。所以vrt文件数据量小，因此vrt文件用于作为中间文件的投影变换结果影像具有很大的优势，可以减少大量IO操作。

[0013] 第四步：计算投影变换结果影像的瓦片最大级别和最小级别，并更新目标瓦片的级别。

[0014] 假设以像素为单位每个瓦片的大小为tilesize×tilesize，投影变换结果影像长

为Xsize、宽为Ysize，以米(m)为单位投影变换结果影像分辨率为reslt。以一张世界地图而言，Resolution(h)表示其第h级别的瓦片分辨率，Resolution(h)可以通过以下公式计算得出Resolution(h)=(2×π×R)/(tilesize×2^h)，R为地球半径等于6378137m，h为整数，π为圆周率。投影变换结果影像的瓦片最大级别tmaxz为Resolution(h)最接近reslt但不大于reslt时的h值；假设minRes=(reslt×max(Xsize,Ysize))/tilesize，max为取最大值函数，则瓦片最小级别tminz为Resolution(h)最接近minRes但不大于minRes时的h值。如果之前设定的目标瓦片级别level>tmaxz则将tmaxz的值赋给level，如果level<tminz则将tminz的值赋给level。而后在瓦片输出目录下创建名为level的文件夹。

[0015] 第五步：计算投影变换结果影像的瓦片行列号范围，并按照路径为/目标瓦片级别/瓦片列号/瓦片行号.png的路径建立空瓦片文件；

[0016] 计算当前level级别下投影变换结果影像地理范围内瓦片的行列号范围，具体计算方法如下：假设投影变换结果影像地理范围为[ominX,ominY,omaxX,omaxY]，其中该范围为一个矩形四边形，ominX,omaxY为左上角点坐标，omaxX,ominY为右下角点坐标；瓦片行列号范围为[tminX,tminY,tmaxX,tmaxY]，其中该范围为一个矩形四边形，tminX,tmaxY为左上角行列号，tmaxX,tminY为右下角行列号，则其之间满足以下映射关系：

[0017] tminX=ceil((ominX+originShift)/(Resolution×tilesize))-1；

[0018] tminY=ceil((ominY+originShift)/(Resolution×tilesize))-1；

[0019] tmaxX=ceil((omaxX+originShift)/(Resolution×tilesize))-1；

[0020] tmaxY=ceil((omaxY+originShift)/(Resolution×tilesize))-1；

[0021] 其中originShift=(2×π×6378137)/2即地球周长的一半，Resolution=(2×π×6378137)/(tilesize×2^{level})，tilesize为瓦片边长大小，ceil表示向上取整。并按照路径为/level/瓦片列号/瓦片行号.png的路径建立空瓦片文件；

[0022] 第六步：任务划分。

[0023] 任务划分是以瓦片为单位采用车轮法，具体划分方法如下：假设rank(i)表示第i个进程，则属于rank(i)任务池中的瓦片其瓦片行列号[tx,ty]满足tx=tminX+(pos%twidth)；$ty = tmaxY - |(pos/twidth)|$，其中twidth=tmaxX-tminX，theight=tmaxY-tminY，pos=j×n+i，(j∈[0,1…,k])，j为整数，$k = \left\lfloor \frac{tcount}{n} \right\rfloor$，⌊ ⌋表示向下取整，tcount=twidth×theight，n表示进程总数，i为当前进程的进程号，tminX,tminY,tmaxX,tmaxY为第五步所求的瓦片最大最小行列号。如果当tcount%n≠0时，进程号i<tcount%n的部分进程还需处理瓦片行列号[tx_2,ty_2]满足以下条件的瓦片：tx_2=tminX+(pos_2%twidth)；$ty_2 = tmaxY - |(pos_2/twidth)|$，其中$pos_2$=tcount-tcount%n+i。

[0024] 各进程按照第六步任务划分规则，然后根据瓦片行列号按照从上到下从左到右的顺序从任务池中取出瓦片行列号，依次重复进行下面步骤。

[0025] 第七步：读取数据，假设rank(i)表示第i个进程，rank(i)当前处理的瓦片行列号为[tx_i,ty_i]，首先进程rank(i)在先前瓦片输出目录中的level文件夹内检查是否已经存在名为tx_i的文件夹，如果不存在则在level目录下创建名为tx_i的文件夹。然后rank(i)根据瓦片行列号以及瓦片级别反算瓦片对应的地理范围[gminX,gminY,gmaxX,gmaxY]，单位为米，

具体解算过程如下：gminX＝tx_i×Resolution-originShift，gminY＝ty_i×Resolution-originShift，gmaxX＝(tx_i+1)×Resolution-originShift，gmaxY＝(ty_i+1)×Resolution-originShift，其中Resolution＝(2×π×6378137)/(tilesize×2^{level})，originShift＝(2×π×6378137)/2。然后求解瓦片地理范围与投影变换结果影像地理范围相交的区域，假设相交区域为[gistminX，gistminY，gistmaxX，gistmaxY]，单位为米，然后计算相交区域在投影变换结果影像中的相对位置以及大小，具体解算过程如下：以米为单位假设投影变换结果影像的地理范围为[gimgminX，gimgminY，gimgmaxX，gimgmaxY]，空间分辨率为Geores，则 $rxsize = \frac{gistmaxX - gistminX}{Geores}$，$rysize = \frac{gistmaxY - gistminY}{Geores}$ 当gistminX＝gimgminX时rx＝0，否则 $rx = \frac{gistminX - gimgminX}{Geores}$；当gistmaxY＝gimgmaxY时ry＝0，否则 $ry = \frac{gimgmaxY - gistmaxY}{Geores}$；其中rx，ry为投影变换结果影像中以左上角为原点的像素坐标系中的读取位置，rxsize和rysize为读取大小。而后利用GDAL类库中的RasterIO函数将rx，ry，rxsize，rysize填入相应参数位置，将相交区域的原始栅格数据读入rank(i)的内存空间中。

[0026] 第八步：重采样，并输出瓦片。

[0027] 由于瓦片的分辨率不一定与影像分辨率一致，因此需要将第七步各进程读入的数据重采样到瓦片相应的分辨率下，重采样方法采用最邻近内插法，重采样后数据大小wxsize，wysize计算方法如下：$wxsize = \frac{rxsize \times tilesize}{(gmaxX - gminX)/Geores}$，$wysize = \frac{rysize \times tilesize}{(gmaxY - gminY)/Geores}$。利用最邻近内插法将栅格数据大小从rxsize×rysize重采样到wxsize×wysize。

[0028] 根据之前假设，rank(i)当前处理的瓦片行列号为[tx_i，ty_i]，首先进程rank(i)在先前的tx_i文件夹目录内检查是否已经存在名为ty_i的瓦片，如果存在则跳过该瓦片，回到第七步，从任务池中取出下一个瓦片的行列号，进行下一个瓦片的处理；如果不存在则rank(i)利用GDAL类库的CreateCopy函数创建一个空的瓦片文件，解算重采样数据在瓦片文件中的写入位置(wx，wy)，计算方法如下：当gistminX＝gimgminX时 $wx = \frac{gistminX - gminX}{gmaxX - gminX} \times tilesize$，否则wx＝0；当gistmaxY＝gimgmaxY时 $wy = \frac{gmaxY - gistmaxY}{gmaxY - gmaxY} \times tilesize$，否则wy＝0。然后利用GDAL的RasterIO函数将wx，wy，wxsize，wysize填入相应参数将重采样数据写入创建好的空瓦片文件中。如果该进程任务池中还有瓦片任务，则返回第七步依次进行下一个瓦片的切片工作，如果没有则该进程切片任务结束。

[0029] 本发明的有益效果是：

[0030] (1)本发明所切出来的瓦片与在目录结构与文件命名上与ArcGIS切出来的保持一致，能够直接应用到Google Map，Mapnik等互联网地图应用中。并且本发明配置简单，可以快速集成到各大WebGIS以及互联网地图应用中。

[0031] (2)本发明切片效率高。算法效率可以很轻松达到存储介质的最大读写带宽，支持多节点并行切片，摆脱了传统单节点切片方法，硬件资源受限且资源利用率低下的问题。并

且所需硬件条件简单，成本低廉，无需GPU等额外计算资源，可以很方便地将一些闲置或淘汰的计算资源重新组织起来，进而结合成一个高性能的切片集群服务器。

[0032] (3)本发明的算法并行程度高加速比明显。各进程之间切片工作相互独立，互不影响，并且横向扩展性能好，随着计算资源或者存储介质读写带宽的增加，算法性能会有很明显提升。

[0033] (4)本发明算法稳定性高。支持切分各种主流数据类型的栅格数据，内存利用率高，支持切分大数据量影像，支持断点功能，可以避免例如断电等外部因素对切片工作带来的干扰。

附图说明

[0034] 图1是本发明算法整体流程图；

[0035] 图2是本发明瓦片切分模型原理示意图；

[0036] 图3是本发明各进程任务划分实施例示意图；

[0037] 图4是本发明影像切片过程示意图；

[0038] 图5是本发明与ArcGIS对不同大小的栅格影像进行切片的性能对比；

[0039] 图6是本发明针对各种规模影像进行切片时执行时间随进程数目变化情况。

具体实施方式

[0040] 为了使本发明的原理，实施步骤，以及优势阐述的更加清楚明白，以下结合附图及相关实例，对本发明进行进一步详述。

[0041] 图1为本发明的整体流程示意图，首先设置主进程、目标瓦片级别level、进程总数n以及瓦片输出根目录，而后主进程读取原始栅格影像投影坐标、长宽等元数据信息，主进程利用GDAL(Geospatial Data Abstraction Library，地理空间数据抽象库)类库的GDALAutoCreateWarpedVRT函数将原始栅格影像投影变换到WebMercator投影坐标系下，投影变换结果影像以vrt格式文件进行保存，后续的所有切片操作都是基于投影变换结果影像进行，其余进程阻塞直到主进程将投影变换结果影像生成完毕为止。主进程投影变换结果影像生成完毕后，各进程同时打开投影变换结果影像，根据影像的空间分辨率计算其所能切出瓦片的最大级别tmaxz和最小级别tminz，判断当前设置的目标瓦片级别level是否大于最大级别tmaxz或小于最小级别tminz，如果大于最大级别tmaxz则将tmaxz重新赋值给level，如果小于最小级别tminz则将tminz重新赋值给level。各进程计算当前level级别下投影变换结果影像地理范围内所覆盖的瓦片行列号范围。根据瓦片行列号范围，采用车轮法为每个进程分配具体处理的瓦片的行列号，所有待处理的瓦片根据瓦片的行列号按照从上到下、从左到右的顺序构成该进程的任务池。各进程从任务池中依次取出瓦片的行列号，根据瓦片行列号以及级别在瓦片输出路径下创建如下目录以及空瓦片文件：“root/level/tx/ty.png”，其中root为瓦片输出根目录，level为目标瓦片级别，tx为瓦片列号，ty为瓦片行号，字符串“root/level/tx/ty.png”作为瓦片的唯一编码，浏览器前端可以通过这个编码对应的URL(Uniform Resource Locator，统一资源定位符)直接访问相应的瓦片数据。空瓦片文件创建完毕后，各进程根据瓦片级别level反算该行列号的瓦片其对应的地理坐标范围，根据地理范围求解其与投影变换结果影像的相交区域，计算相交区域相对于投影变

换结果影像中的像素坐标以及范围，而后根据像素坐标及范围利用GDAL类库的RasterIO函数将该部分相交区域数据从投影变换结果影像中读取到该进程的内存空间中。随后各进程将相交区域数据从投影变换结果影像的分辨率下重采样到瓦片level分辨率下，最后将重采样数据写入之前创建好的空瓦片文件中，则当前瓦片生成完毕。如果进程的任务池中还有瓦片未曾处理，则依次对下一个瓦片进行处理，否则该进程瓦片切分任务完成。

[0042] 图2为本发明瓦片切分模型原理示意图，示意图以一张世界地图为例，切分规则基于国际开源地理空间基金组织提出的TMS(Tile Map Service，瓦片地图服务)协议，地图投影采用WebMercator(Web墨卡托投影)。假设地球被套在一个圆柱中，赤道与圆柱相切，然后在地球中心放一盏灯，把球面上的图形投影到圆柱体上，再把圆柱体展开，这就形成了一幅墨卡托投影的世界地图，其原点在经纬度(0,0)处。由于理论上南北极是永远无法投影到圆柱体上，并且随着纬度的增高其变形越大，为了方便，web墨卡托投影忽略了墨卡托投影中南北两级变形较大的区域，把椭圆形的地球投影成平面上边长等于赤道周长的正方形，其大地坐标范围为[-180°,-85.0511287798°,180°,85.0511287798°]，投影坐标范围为[-20037508.3427892m,-20037508.3427892m,20037508.3427892m,20037508.3427892m]，瓦片切分则是基于这个投影坐标系统，将栅格影像进行不同分辨率的切分。以一幅世界地图栅格影像为例，level表示瓦片的级别，tileszie为瓦片的长宽，则每一级别共有4^{level}个tileszie×tileszie大小的瓦片，因此每个级别瓦片对应的空间分辨率Resolution=(2×20037508.3427892)/(tilesize×2^{level})，瓦片划分采用四叉树的方式进行，即以赤道和本初子午线的交点作为中心，不断对地图进行四分，直到每个格网大小为tilesize×tilesize为止。如图2(a)的图所示，0级世界地图由一个瓦片表示，1级世界地图应由4个瓦片表示，2级16个，以此类推。因此当对一个普通栅格影像进行切片时，首先根据栅格影像的分辨率找到与其分辨率最接近的瓦片级别，而后通过上述所述的世界地图切分规则，计算影像所在该层世界地图瓦片中的位置，进行切片。当地图被划分为4^{level}个瓦片后，需要对瓦片进行编码，存储在文件系统的相应区域以便浏览器前端能够快速访问，瓦片编码以瓦片在瓦片坐标系内的行列号为基础进行编制而成，如图2(b)的图所示，瓦片坐标系的原点位于左上角，瓦片存储在文件系统内分为三级目录其中第一级为瓦片级别(level)，第二级为瓦片列号(tx)，第三级为瓦片行号(ty)。以字符串“root/level/tx/ty.png”作为瓦片图片的唯一编码，只需要正确解析瓦片的编码就可以获取地图瓦片相应的访问路径。

[0043] 图3是本发明各进程任务划分实施例示意图，其描述的是一个以瓦片为单位大小为5×4的栅格影像被8个进程进行切分的任务划分示例，该任务划分方法名为车轮法，其原理就是各进程按照进程号从小到大的顺序，根据瓦片坐标系下瓦片的分布，从左到右，从上到下依次进行分配，当一个周期完毕后，接着上一周期的位置重复进行上述类似操作，直至各进程将所有瓦片分配完毕。可以通过如下公式计算得到每个进程所分配得到的瓦片的行列号：假设进程的进程号为i，则进程i分配得到的瓦片其瓦片行列号集合，集合中元素[tx,ty]均满足tx=tminX+(pos%twidth)；$ty = tmaxY - [(pos / twidth)]$，其中twidth=tmaxX-tminX，theight=tmaxY-tminY，pos=j×n+i，j∈[0,1…,k]，$k = \left[\frac{tcount}{n}\right]$，tcount=twidth×theight，n表示进程总数，i为当前进程的进程号，tminX，tminY，tmaxX，tmaxY为第五步所求的瓦片最大最小行列号。如果当tcount%n≠0时，进程号i<tcount%n

的部分进程还需处理瓦片行列号[tx_2,ty_2]满足以下条件的瓦片：tx_2=tminX+(pos_2%twidth)；$ty_2 = tmaxY - [(pos_2/twidth)]$，其中$pos_2$=tcount-tcount%n+i。

[0044] 对照图3，将具体数值融入公式后对相关原理进行叙述，便于理解本发明步骤。首先拿到投影变换结果影像后，根据其地理范围解算与其范围有相交的瓦片，因为每个行列号所对应的瓦片其地理范围是一定的，因此得到相交瓦片行列号的取值范围，本发明最终目的就是要将影像切割成一小块一小块的瓦片，图3将投影变换结果影像切割成4行5列共20块瓦片，即tcount=20。并假设解算出来的瓦片列号tx范围为0～4，ty范围为0～3，进程总数n=8，则$k=[tcount/n]=2$，因无法整除，表示所有进程在执行完两轮分配后，进程号i<tcount%n的部分进程即进程0、进程1、进程2、进程3要处理剩下的$(tcount-n\times[tcount/n]=20-8\times2)=4$个瓦片，8个进程总共要执行3轮，才能把所有瓦片任务分配完，而前两轮所有进程都要参与，第三轮部分进程参与。以进程0为例，循环3轮，每轮中分配到一个瓦片，在图中按照瓦片行列号从上到下从左到右的顺序分别对应行列号坐标为(0,3)、(3,2)、(1,0)的瓦片，将这些瓦片按照顺序依次加入该进程的任务池中，瓦片在任务池中的顺序即序列号j，即(0,3)的序列号为0，(3,2)的序列号为1，(1,0)的序列号为2。为了通过瓦片序列号j和进程号i反算出对应瓦片的行列号，先一个中间变量pos，以进程0的三个瓦片为例(0,3)号瓦片的pos等于0×8+0=0，(3,2)号瓦片的pos等于1×8+0=8，(1,0)号瓦片是针对tcount%n≠0不等于0的情况，等于20-20%8+0=16。最后通过pos就可以反算瓦片的行列号了，比如(3,2)号瓦片的列号tx就可以这么计算tx=(0+8%5=3)，瓦片行号$ty=(3-[8/5]=2)$；(1,0)号瓦片tx=(0+16%5=1)，$ty=(3-[16/5]=0)$。这样就将任务池中瓦片的序列号和瓦片的行列号映射了起来，这样各进程只需遍历j就可以依次处理对应行列号的瓦片。

[0045] 图4是本发明影像切片过程示意图，首先根据瓦片行列号以及瓦片级别解算瓦片的地理范围，下面以level为10级下的(100,110)号瓦片为例阐释解算方法：(100,110)号瓦片的行号ty=110，列号tx=100，假设瓦片大小为256×256(单位为像素)，其瓦片地理范围为[gminX,gminY,gmaxX,gmaxY]那么gminX=100×Resolution-originShift=-20022220.59gminY=110×Resolution-originShift=-20020689同理gmaxX=(100+1)×Resolution-originShift=-20022065.1gmaxY=(110+1)×Resolution-originShift=-20020536.1)，其中Resolution=(2×π×6378137)/(256×2^{10})=152.9，originShift=(2×π×6378137)/2=20037508，单位为m，然后求解瓦片地理范围与投影变换结果影像地理范围的交集，假设影像地理范围为[-20022110.59,-20021579,-20021085,-20020836]，分辨率为15m；那么相交部分区域地理范围为[-20022110.59,-20020689,-20022065.1,-20020836])，将根据地理范围将相交区域数据从投影变换结果影像中取出，将相交数据从投影变换结果影像的分辨率下重采样到level级别下瓦片的分辨率(即从15m分辨率重采样到152.9米)，然后根据相交区域地理范围解算其在瓦片文件中的相对位置，最后将重采样数据写入空的瓦片文件相应区域。

[0046] 图5是本发明与当今主流商业软件ArcGIS对不同大小的栅格影像进行切片的性能对比，在相同硬件条件下，所采用栅格影像数据为1.tif、2.tif、3.tif，其中数据数据详细信息见下表：

[0047]

影像名称	影像长	影像宽	数据类型	波段数	大小	分辨率	最大切片级别	投影信息
1.tif	17152	11520	8位字节	3	605.04MB	2.389米	15	WebMercator
2.tif	38265	38376	8位字节	3	4.62GB	0.00014度	13	无（大地坐标系）
3.tif	87040	58368	8位字节	3	15.2G	0.299米	19	WebMercator

[0048] 切片级别都采用影像的最大切片级别，并行切片算法进程总数设置为16，由于ArcGIS支持多线程加速，为了保证实验可对比性ArcGIS切片参数中设置最大线程数16，图中纵坐标为算法耗时(单位为秒)，横坐标为测试影像名称，表格内数字表示相应算法对应不同数据的耗时(单位为秒)，由于ArcGIS不支持对非WebMercator投影的影像直接切分成WebMercator投影系下的瓦片，因此对应2.tif的ArcGIS实验数据无法获得，从这也能看到当前商业软件切片方法的复杂与局限性。可以看到本发明在面对不同大小的栅格影像都保持稳定高效的效率，具有良好的线性加速比，特别是随着影像数据量以及分辨率的提高，这种优势相对ArcGIS来说更加明显。针对图中2.tif数据量比3.tif小，而切片时间却更长的原因主要在于2.tif其坐标系统是大地坐标无投影信息，因此需要将影像从大地坐标系投影变换到WebMercator投影坐标系，而3.tif本身为WebMercator投影因此无需进行投影变换，所以这直接影响到了加速性能。

[0049] 图6是本发明针对各种规模影像进行切片时执行时间随进程数目变化情况。实验数据采用上述所述的1.tif，2.tif，3.tif三幅影像，切片级别采用影像最大切片级别，横坐标是进程数，纵坐标是算法执行时间(单位为秒)，表格内数字表示相应算法对应不同数据的耗时(单位为秒)。从图中可以看到：(1)一定范围内随着进程数的增加，本发明的效率逐步提升，但是随着进程数的持续增加，算法耗时逐步趋于某一稳定值，如果进程数继续加大，算法速度在某一程度反而会出现下降；(2)随着数据量的增大，算法的并行化效率愈加明显，并且趋于算法耗时稳定值的进程数也相应更大。其原因如下：随着进程数的增加，任务被划分给更多进程执行，每个进程对应任务规模相应更小，所以算法整体性能得到提升。当进程数目增大到一定程度后，受限于硬盘的读写速度，算法性能趋于稳定，但是如果继续增大那么可能出现各进程读写竞争的情况，导致性能下降。所以针对不同规模的影像，应采用相应合理的进程数才能获得最优的执行效率，通过测试表明该发明针对不同规模的影像具有良好的扩展性。

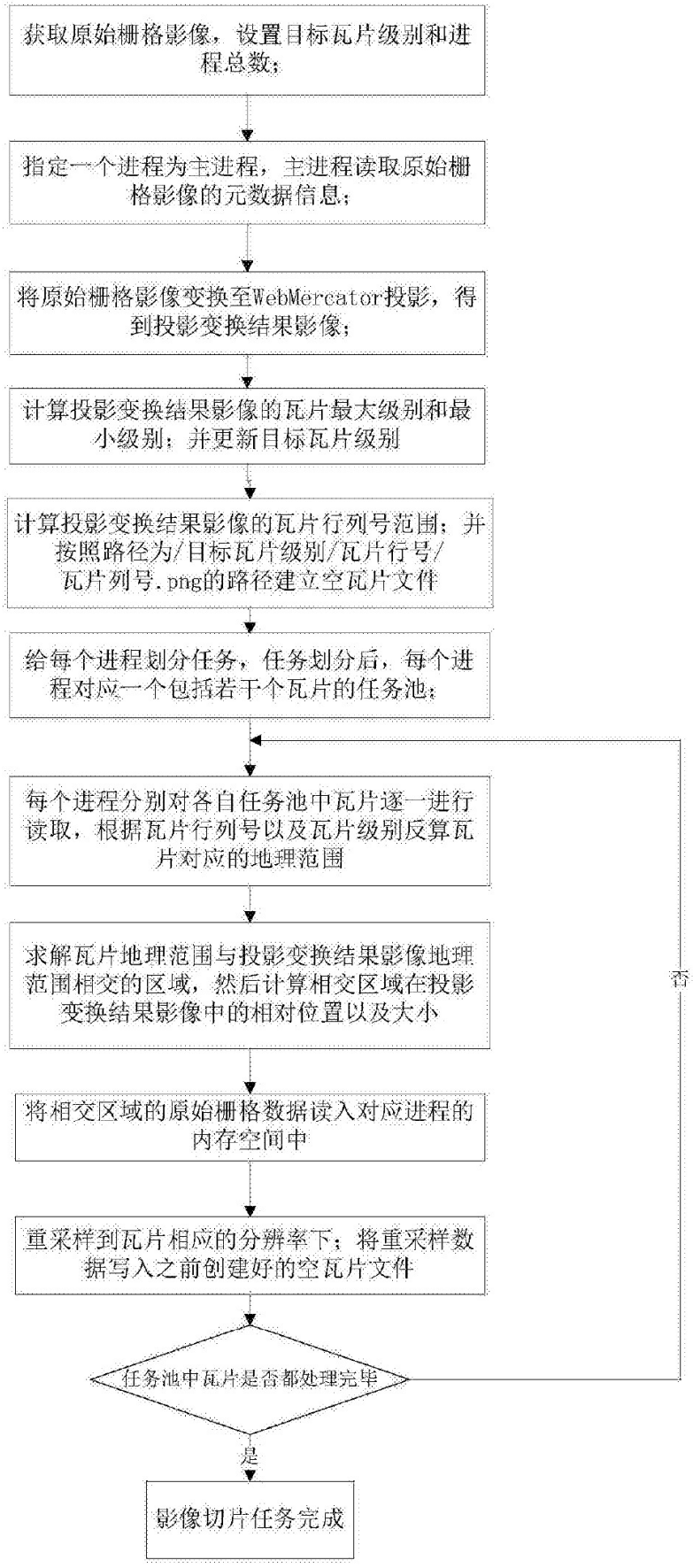
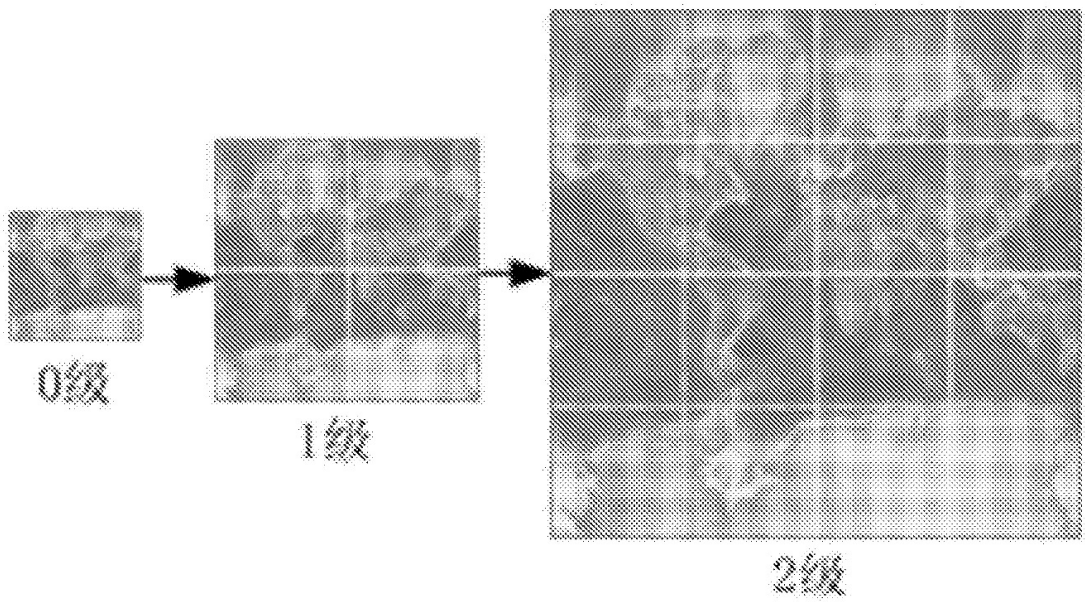
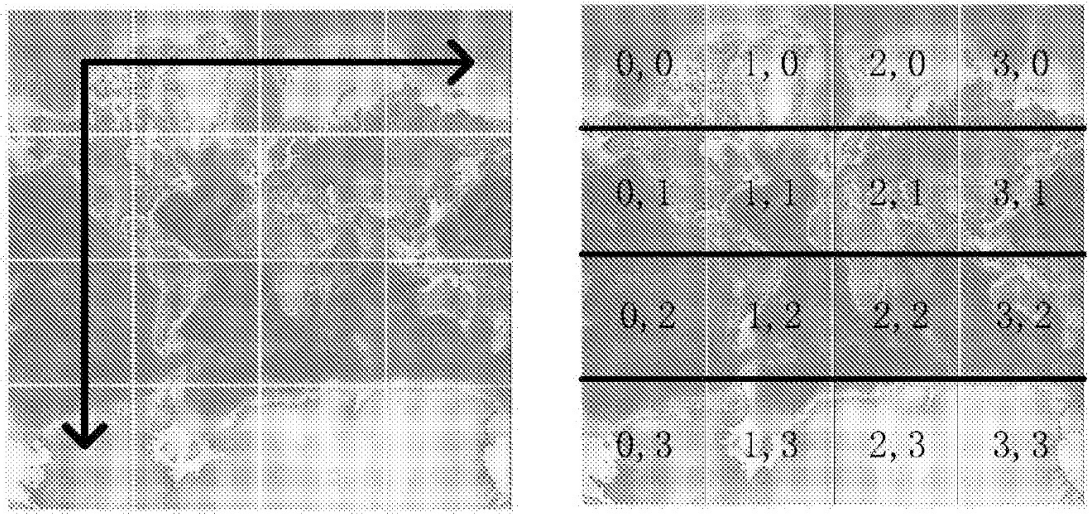

获取原始栅格影像，设置目标瓦片级别和进程总数；
指定一个进程为主进程，主进程读取原始栅格影像的元数据信息；
将原始栅格影像变换至WebMercator投影，得到投影变换结果影像；
计算投影变换结果影像的瓦片最大级别和最小级别；并更新目标瓦片级别
计算投影变换结果影像的瓦片行列号范围；并按照路径为/目标瓦片级别/瓦片行号/瓦片列号.png的路径建立空瓦片文件
给每个进程划分任务，任务划分后，每个进程对应一个包括若干个瓦片的任务池；
每个进程分别对各自任务池中瓦片逐一进行读取，根据瓦片行列号以及瓦片级别反算瓦片对应的地理范围
求解瓦片地理范围与投影变换结果影像地理范围相交的区域，然后计算相交区域在投影变换结果影像中的相对位置以及大小
将相交区域的原始栅格数据读入对应进程的内存空间中
重采样到瓦片相应的分辨率下；将重采样数据写入之前创建好的空瓦片文件
任务池中瓦片是否都处理完毕
否
是
影像切片任务完成

图1



(a)



(b)

图2

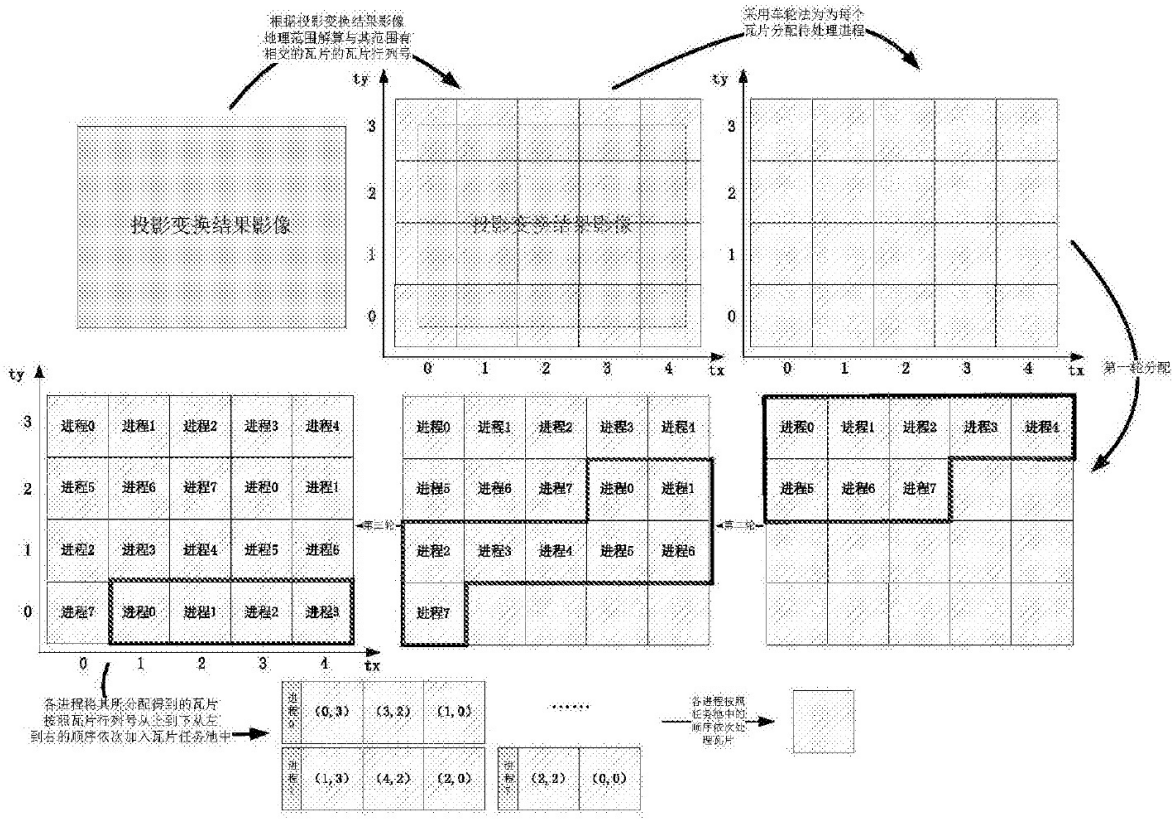

根据投影变换结果影像地理范围解算与其范围有相交的瓦片的瓦片行列号
采用车轮法为为每个瓦片分配待处理进程
投影变换结果影像
投影变换结果影像
ty
tx
第一轮分配
第二轮
第三轮
进程0
进程1
进程2
进程3
进程4
进程5
进程6
进程7
各进程将其所分配得到的瓦片按照瓦片行列号从左到下从左到右的顺序依次加入瓦片任务池中
(0,3)
(3,2)
(1,0)
(1,3)
(4,2)
(2,0)
(2,2)
(0,0)
各进程按照任务池中的顺序依次处理瓦片

图3

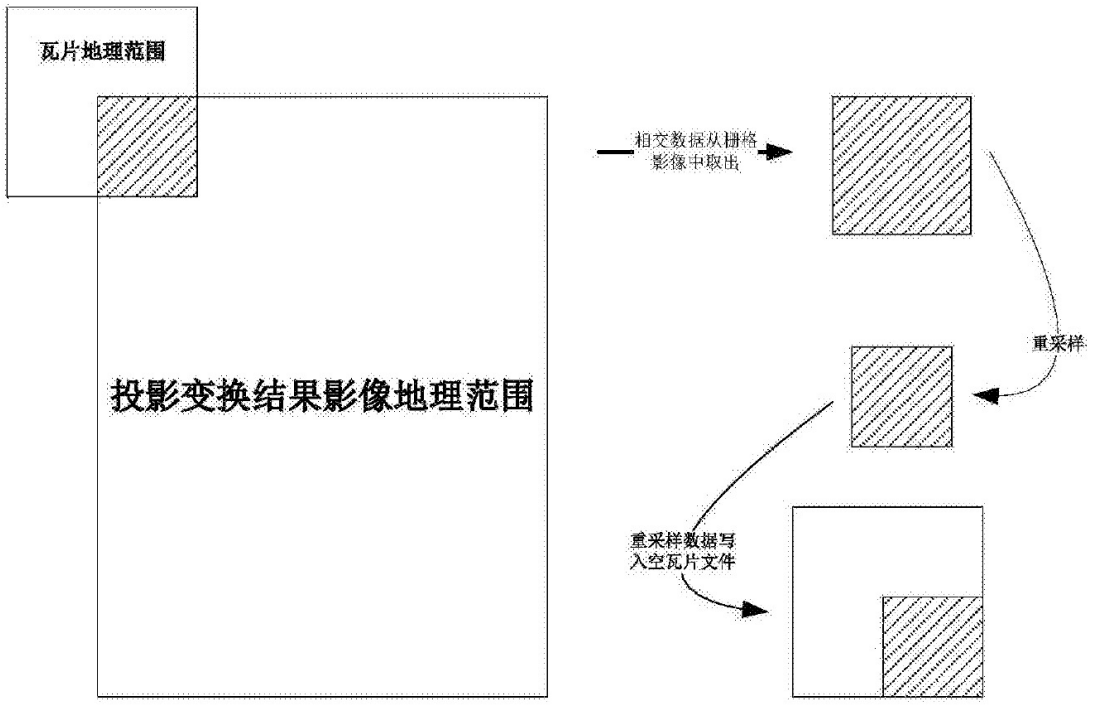
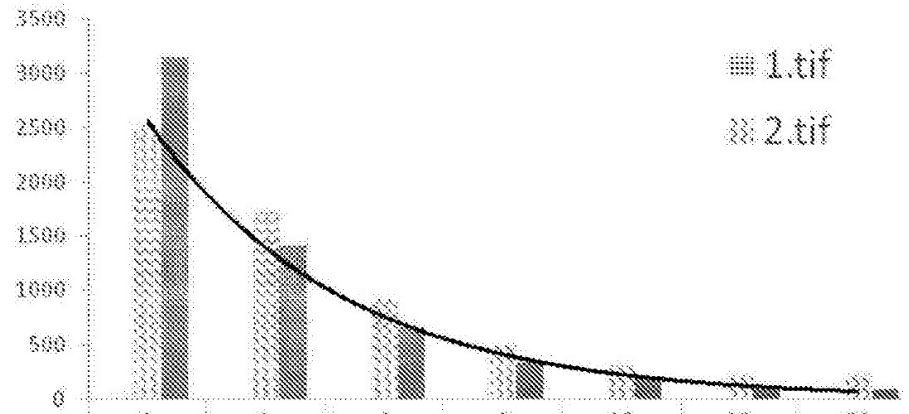

瓦片地理范围
投影变换结果影像地理范围
相交数据从栅格影像中取出
重采样
重采样数据写入空瓦片文件

图4

	1.tif	2.tif	3.tif
ArcGIS	54.18		1795
本发明	3.32	319.11	198.02

图5



	1	2	4	8	16	32	64
1.tif	23.66	14.16	7.22	3.93	3.32	1.65	1.73
2.tif	2527.33	1731.56	925.19	498.5	319.11	231.47	224.73
3.tif	3148.15	1416.9	660.06	336	198.02	118.39	81.39

图6